

# Package ‘cartogramR’

October 18, 2022

**Version** 1.0-9

**Date** 2022-10-18

**Title** Continuous Cartogram

**Description** Procedures for making continuous cartogram. Procedures available are:  
flow based cartogram (Gastner & Newman (2004) <[doi:10.1073/pnas.0400280101](https://doi.org/10.1073/pnas.0400280101)>),  
fast flow based cartogram (Gastner, Seguy & More (2018) <[doi:10.1073/pnas.1712674115](https://doi.org/10.1073/pnas.1712674115)>),  
rubber band based cartogram (Dougenik et al. (1985)  
<[doi:10.1111/j.0033-0124.1985.00075.x](https://doi.org/10.1111/j.0033-0124.1985.00075.x)>).

**Depends** R (>= 3.5.0)

**Imports** sf, data.table

**Suggests** lwgeom

**SystemRequirements** FFTW (>=3.3.1); possible package: fftw-devel (rpm),  
libfftw3-dev (deb) or fftw (brew).

**License** MIT + file LICENSE

**RoxygenNote** 7.2.0

**Encoding** UTF-8

**NeedsCompilation** yes

**Author** Pierre-Andre Cornillon [aut, cre],  
Florent Demoraes [aut],  
Flow-Based-Cartograms [cph] (Author of core C code for gsm and gn  
procedures)

**Maintainer** Pierre-Andre Cornillon <[pierre-andre.cornillon@univ-rennes2.fr](mailto:pierre-andre.cornillon@univ-rennes2.fr)>

**Repository** CRAN

**Date/Publication** 2022-10-18 16:55:07 UTC

## R topics documented:

as.sf . . . . .	2
as.sf.cartogramR . . . . .	3
as.sfc . . . . .	3
as.sfc.cartogramR . . . . .	4

as.sfmultipolygon . . . . .	4
cartogramR . . . . .	5
cartogramR_options . . . . .	6
check_ring_dir . . . . .	8
dist_between_vertices . . . . .	9
france_dept . . . . .	9
france_rivers . . . . .	10
from_coord_polygon . . . . .	10
geom_cartogramR . . . . .	11
grid_analysis . . . . .	12
make_layer . . . . .	12
plot.cartogramR . . . . .	13
plot.dbv.cartogramR . . . . .	14
plot.gridanalysis.cartogramR . . . . .	15
precartogramR . . . . .	16
print.cartogramR . . . . .	17
print.summary.cartogramR . . . . .	18
residuals.cartogramR . . . . .	18
summary.cartogramR . . . . .	19
summary.dbv.cartogramR . . . . .	20
summary.gridanalysis.cartogramR . . . . .	21
to_coord_polygon . . . . .	22
usa . . . . .	22

## Index 24

---

as.sf	<i>Coerce an object to a sf object</i>
-------	--

---

### Description

Coerce an object to a sf object

### Usage

```
as.sf(x, ...)
```

### Arguments

x	object to be coerced
\dots	arguments passed to or from other methods.

### Value

an sf object

---

as.sf.cartogramR	<i>Coerce a cartogramR to a sf object</i>
------------------	---

---

**Description**

Coerce a cartogramR to a sf object returning the sf object used to construct the cartogram with the cartogram as geometry and some more attributes

**Usage**

```
## S3 method for class 'cartogramR'  
as.sf(x, ...)
```

**Arguments**

x	a cartogramR object
\dots	arguments passed to or from other methods.

**Value**

a sf object including all the data (attributes) contained in the original sf object used to construct the cartogram and

- original areas of region (orig\_area)
- final/deformed areas of region (final\_area)
- target areas of region (target\_area)
- original centers (x\_orig\_centers and y\_orig\_centers)
- final centers (x\_final\_centers and y\_final\_centers)

---

as.sfc	<i>Coerce an object to a sfc object</i>
--------	---

---

**Description**

Coerce an object to a sfc object

**Usage**

```
as.sfc(x, ...)
```

**Arguments**

x	object to be coerced
\dots	arguments passed to or from other methods.

**Value**

a sfc object

---

as.sfc.cartogramR      *Coerce a cartogramR to a sfc object*

---

**Description**

Coerce a cartogramR to a sfc object extracting the component cartogram of the cartogramR object

**Usage**

```
## S3 method for class 'cartogramR'
as.sfc(x, ...)
```

**Arguments**

x                      a cartogramR object  
\dots                    arguments passed to or from other methods.

**Value**

a sfc object

---

as.sfmultipolygon      *Transform a sf object with several rows (polygons) by region to an sf object with one row by region and thus one multipolygon by region*

---

**Description**

Transform a sf object with several rows (polygons) by region to an sf object with one row by region and thus one multipolygon by region

**Usage**

```
as.sfmultipolygon(data, idregion, closepolygon = FALSE)
```

**Arguments**

data                    a sf object  
idregion                a character string which indicates the name of the column (in data object) which contains the region identifier.  
closepolygon          a boolean (default to FALSE) if TRUE it controls if polygons are closed and if not add the first vertice at the end.

**Value**

a sf object with one row by region and one multipolygon by region.

---

cartogramR                      *Make a continuous cartogram (density equalizing maps)*

---

## Description

Make a continuous cartogram (density equalizing maps)

## Usage

```
cartogramR(  
  data,  
  count,  
  method = c("gsm", "gn", "dcn", "GastnerSeguyMore", "GastnerNewman",  
             "DougenikChrismanNiemeyer"),  
  options = NULL  
)
```

## Arguments

data	a sf object which contains at least two columns: obviously a geometry column (giving the map) and a column which contains a count by region (leading to a density by region, density to be equalized by deformation). Each row of data is a region and contains the simple feature geometry of type POLYGON or MULTIPOLYGON. Polygon ring directions are not checked but exterior ring must counter clockwise and holes clockwise (use option <code>check_ring_dir</code> of <code>sf::st_read</code> to achieve the right orientation of ring direction on import or use <code>check_ring_dir</code> function)
count	a character string which indicates the name of the column (in data object) which contains the count by region.
method	the method to be used, can be one of the following: <code>gsm</code> or <code>GastnerSeguyMore</code> (default), <code>gn</code> or <code>GastnerNewman</code> , <code>dcn</code> or <code>DougenikChrismanNiemeyer</code> .
options	a named list given to <code>cartogramR_options</code> function which process options see <code>cartogramR_options</code> for details. Default to <code>NULL</code> .

## Value

A list with the following components:

- `cartogram`: a sf object (in the same order of data or sorted by `idregion` see `reordered` argument) which contains the cartogram (ie the initial polygons after deformation)
- `orig_area`: original areas of regions
- `final_area`: final areas of regions in the cartogram
- `orig_centers`: the initial centers calculated with `st_point_on_surface`
- `final_centers`: the centers after deformation

- `gridx`: (for flow-based method) final grid (x-axis) if requested (see [cartogramR\\_options](#) for details).
- `gridy`: (for flow-based method) final grid (y-axis) if requested (see [cartogramR\\_options](#) for details).
- `count`: the count by region
- `target_area`: target areas of regions
- `initial_data`: the initial sf object
- `details`: names of original data, idcount variable, algorithm
- `options`: values of options

## References

- Dougenik, J., Chrisman, R. & Niemeyer, D. (1985). An algorithm to construct continuous area cartograms. *Professional Geographer* **37**: 75-81.
- Gastner, M. & Newman, M.E.J. (2004). Diffusion-based method for producing density equalizing maps. *Proc. Natl. Acad. Sci. USA*, **101**:7499-7504
- Gastner, M., Seguy, V. & More, P. (2018). Fast flow-based algorithm for creating density-equalizing map projections. *Proceedings of the National Academy of Sciences USA*, **115**:E2156-E2164, website: [go-cart](#)

## Examples

```
data(usa)
carto <- cartogramR(usa, "electors64")
plot(carto)
summary(carto)
```

---

`cartogramR_options`      *Set the options of [cartogramR](#) in the correct format*

---

## Description

Set the options of [cartogramR](#) in the correct format

## Usage

```
cartogramR_options(  
  options,  
  method = c("gsm", "gn", "dcn", "GastnerSeguyMore", "GastnerNewman",  
             "DougenikChrismanNiemeyer")  
)
```

## Arguments

- options a named list with some (or all) the following components:
- maxit: (all method) the maximum number of iterations, default to 50.
  - absrel: (all method) boolean, if TRUE relative convergence if FALSE absolute convergence (default to TRUE)
  - abserror: (all method) Areas on cartogram differ at most by an (absolute value of) error of abserror. That is,  $\max\_polygons \text{ larea\_on\_cartogram} - \text{target\_areal} \leq \text{abserror}$  (default to 10000)
  - abstol: ("dcn") the absolute convergence error tolerance:  $\max\_polygons \text{ larea}(i) - \text{area}(i-1)$  default to 1000
  - relerror: (all method) Areas on cartogram differ at most by an (absolute value of) relative error of relerror. That is,  $\max\_polygons \text{ larea\_on\_cartogram} / \text{target\_area} - 1 \leq \text{relerror}$  (default to 0.01)
  - reltol: ("dcn") the absolute convergence tolerance:  $\max\_polygons \text{ abs}((\text{area}(i) - \text{area}(i-1))/\text{area}(i-1))$  default to  $1e-3$
  - L: ("gsm" or "gn") integer, gives the value of L (default is 512), must be a power of two (for fftw)
  - mp: (all method) if a region contains exactly zero population, it will be replaced by mp times the smallest (strictly) positive population in any region (default to 0.2)
  - pf: ("gsm" or "gn") Determines space between map and boundary (default to 1.5)
  - sigma: ("gsm" or "gn") Width of Gaussian blur to smoothen the density (default to 5)
  - center: ("gsm" or "gn") either a character string (only possible choices are "centroid" or "point\_on\_surface") or a function. If the object is a function, it will be used to calculate the "center" of polygons; "point\_on\_surface" will use the function [sf::st\\_point\\_on\\_surface](#) while "centroid" (the default) will use [sf::st\\_centroid](#).
  - verbose: (all method) integer giving the verbosity level (default to 0, not verbose)
  - grid: ("gsm" or "gn") boolean, if TRUE export the final grid from flow algorithm (default to TRUE). Setting to FALSE
  - check.ring.dir: (all method) boolean, if TRUE controls polygons orientation (default to TRUE)
  - check.only: (all method) boolean, if TRUE control only polygons orientation and no replacement is done (default to FALSE)
- method the method to be used, can be one of the following: gsm or GastnerSeguyMore (default), gn or GastnerNewman, dcn or DougenikChrismanNiemeyer.

## Value

a list to be processed by [cartogramR](#)

## References

- Dougenik, J., Chrisman, R. & Niemeyer, D. (1985). An algorithm to construct continuous area cartograms. *Professional Geographer* **37**: 75-81.
- Gastner, M. & Newman, M. E. J. (2004). Diffusion-based method for producing density equalizing maps. *Proc. Natl. Acad. Sci. USA*, **101**:7499-7504
- Gastner, M., Seguy, V. & More, P. (2018). Fast flow-based algorithm for creating density-equalizing map projections. *Proceedings of the National Academy of Sciences USA*, **115**:E2156-E2164

## Examples

```
data(usa)
carto1 <- cartogramR(usa, "electors64", options=list(verbose=1, L=256))
plot(carto1)
```

---

check\_ring\_dir

*Polygon rings directions are checked and corrected if asked.*

---

## Description

Polygon ring are seen from above: exterior ring counter clockwise, holes clockwise

## Usage

```
check_ring_dir(polygons, check.only = TRUE)
```

## Arguments

polygons	a sfc object which contains simple feature geometry of types POLYGON or MULTIPOLYGON
check.only	a boolean which indicates if the function only checks the ring direction (check.only=TRUE) or checks and corrects the polygon direction (check.only=FALSE)

## Value

Either a logical vector which indicates if line i of polygons is in the right direction (TRUE) or not or the corrected sfc object

## Examples

```
data(usa)
all(check_ring_dir(sf::st_geometry(usa), check.only=TRUE))
```



---

dist\_between\_vertices *Analyse some of the grid options*

---

**Description**

Analyse some of the grid options

**Usage**

```
dist_between_vertices(data)
```

**Arguments**

data                    a sf object to be used in cartogram.

**Value**

a 'dbv.cartogramR' object which is a data-table which contains distance between vertices ('dbv') and polygons names ('L1', 'L2', 'L3') inherited from [sf::st\_coordinates]

**Examples**

```
data(usa)
dbv <- dist_between_vertices(data=usa)
summary(dbv)
```

---

france\_dept                    *Map of the population of mainland France (year 2018)*

---

**Description**

This data set is a basemap of mainland France with the population in 2018 (pop2018), the number of physicians in 2018 (n\_physicians), the number of general practitioner in 2018 (n\_gp) and the number of general practitioner for 100000 inhabitants in 2018 (n\_gp\_per100000) in each department (dept\_name or id).

**Usage**

```
data(france_dept)
```

**Format**

A sf object containing 8 columns of data and the geometry Projected CRS: RGF93 / Lambert-93 (EPSG : 2154)

**Source**

[https://www.data.gouv.fr/fr/datasets/admin-express/#\\_](https://www.data.gouv.fr/fr/datasets/admin-express/#_)

**References**

- [https://www.insee.fr/fr/statistiques/2012713#tableau-TCRD\\_004\\_tab1\\_departements](https://www.insee.fr/fr/statistiques/2012713#tableau-TCRD_004_tab1_departements)
- [https://www.insee.fr/fr/statistiques/2012677#tableau-TCRD\\_068\\_tab1\\_departements](https://www.insee.fr/fr/statistiques/2012677#tableau-TCRD_068_tab1_departements)

---

france\_rivers

*Map of french rivers of mainland France*

---

**Description**

This data set is a basemap of mainland french rivers (name or id).

**Usage**

```
data(france_rivers)
```

**Format**

A sf object containing 2 columns of data and the geometry Projected CRS: RGF93 / Lambert-93 (EPSG : 2154)

**Source**

<https://geoservices.ign.fr/telechargement>

---

from\_coord\_polygon

*Transform from coordinates system used in the polygons to coordinates system used in flow based cartogram*

---

**Description**

Apply the mapping from the coordinates system used in the polygons (characterised by the CRS) to the coordinates system used in flow based cartogram

**Usage**

```
from_coord_polygon(coord, carto)
```

**Arguments**

coord	a vector of length 2 or a two columns matrix containing xy coordinates to transform
carto	a cartogramR object

**Value**

a vector of length 2 or a two columns matrix containing xy coordinates in the coordinate systems of polygons used to build the cartogram

**Examples**

```
data(usa)
carto <- cartogramR(usa, "electors64")
from_coord_polygon(c(-0.007, -0.348), carto)
```

---

geom_cartogramR	<i>Apply the deformation used to build a cartogram to a set of simple geometry coordinates</i>
-----------------	--

---

**Description**

Apply the deformation used to build a cartogram to a set of simple geometry coordinates. The resulting simple geometry object can be used to add geometry features on the cartogram.

**Usage**

```
geom_cartogramR(sfgeom, carto, verbose = FALSE)
```

**Arguments**

sfgeom	a sf or a sfc object which contains simple feature geometry of types in the following POINT, MULTIPOINT, LINESTRING, MULTILINESTRING, POLYGON, 'MULTIPOLYGON'
carto	a cartogramR object
verbose	a boolean object to set on verbose mode (default to FALSE)

**Value**

a sf or a sfc object which contains simple feature geometry transformed

**Examples**

```
data(usa)
carto <- cartogramR(usa, "electors64")
LA <- sf::st_sfc(sf::st_point(c(-118.243685, 34.052234)))
sf::st_crs(LA) <- 4326
moregeom <- geom_cartogramR(LA, carto)
plot(carto)
```

```
plot(moregeom, add=TRUE, col=2, pch=15)
```

---

grid_analysis	<i>Analyse some of the grid options</i>
---------------	---

---

### Description

Analyse some of the grid options

### Usage

```
grid_analysis(data, gridpower2 = 8:11, pf = 1.5, verbose = FALSE)
```

### Arguments

data	a sf object to be used in cartogram.
gridpower2	a vector of exponent (to be raised at the power of 2) that gives the log2(size) of the grid (default to '8:11')
pf	Determines space between map and boundary (default to 1.5)
verbose	a boolean object to set on verbose mode (default to 'TRUE')

### Value

a 'gridanalysis.cartogramR' object which is a matrix

### Examples

```
data(usa)
ga <- grid_analysis(data=usa, gridpower2=4:8, verbose=TRUE)
summary(ga)
```

---

make_layer	<i>Make a layer</i>
------------	---------------------

---

### Description

Create a sfc object containing final centers, original centers, centers displacement, original graticule or final graticule.

### Usage

```
make_layer(
  x,
  type = c("final_centers", "original_centers", "centers_translation",
           "final_graticule", "original_graticule")
)
```

**Arguments**

x	a cartogramR object
type	a character string giving the type of layer: - "final_centers": if method is dcn, <a href="#">sf::st_centroid</a> is applied on deformed/cartogram region ; if method is gsm or gn (ie flow based), initial "centers" are calculated and the cartogram deformation is applied on these "centers" giving the final_centers. - "original_centers" if method is dcn, <a href="#">sf::st_centroid</a> is applied on original regions); if method is gsm or gn (ie flow based), initial "centers" are calculated using cartogramR center option see <a href="#">cartogramR_options</a> . - "centers_translation" linestring giving the movement of centers due to the deformation used to have the cartogram - "final_graticule" (method gsm or gn) graticule obtained by the cartogram algorithm - "original_graticule" (method gsm or gn) graticule used by the cartogram algorithm

**Value**

a sfc object

---

plot.cartogramR	<i>Plot a cartogram object</i>
-----------------	--------------------------------

---

**Description**

Plot a cartogram object

**Usage**

```
## S3 method for class 'cartogramR'
plot(x, ...)
```

**Arguments**

x	a cartogram object
\dots	arguments passed to or from other methods.

**Value**

No return value, called for side effects

**Examples**

```
data(usa)
carto <- cartogramR(usa, "electors64")
plot(carto)
```

---

plot.dbv.cartogramR *Plot a dbv.cartogram object*

---

## Description

Plot a dbv.cartogram object

## Usage

```
## S3 method for class 'dbv.cartogramR'  
plot(x, which = 1:2, ask = TRUE, key = TRUE, last = 10, probminx = 0.9, ...)
```

## Arguments

x	a dbv.cartogram object
which	if a subset of the plots is required, specify a subset of the numbers 1:2
ask	logical; if TRUE, the user is asked before each plot, see <a href="#">par(ask=.)</a>
key	logical; if TRUE, a legend is drawn
last	draw the density of distance between vertices for the last coordinates
probminx	the sample quantiles (of distance between vertices) corresponding to the probability is used as a minimum of x-axis for the density plot (used only if last is NULL)
\dots	arguments passed to or from other methods.

## Details

The first plot is the density of distance between consecutive vertice by region. Only the upper quantiles are shown. The second plot is a barplot by region of the number of vertice divided by the perimeter of the region

## Value

No return value, called for side effects

## Examples

```
data(usa)  
precarto <- precartogramR(usa, method="dcn")  
plot(precarto)
```

---

`plot.gridanalysis.cartogramR`*Plot a gridanalysis.cartogram object*

---

## Description

Plot a gridanalysis.cartogram object

## Usage

```
## S3 method for class 'gridanalysis.cartogramR'
plot(
  x,
  nthsmallest = 5,
  redrawxaxis = TRUE,
  type = "b",
  xlab = NULL,
  ylab = NULL,
  ylim = c(0, 20),
  ...
)
```

## Arguments

<code>x</code>	a gridanalysis.cartogram object
<code>nthsmallest</code>	plot only the nthsmallest values among all polygons
<code>redrawxaxis</code>	if TRUE redraw ticks and labels of x axe at grid size on log scale
<code>type</code>	character string (length 1 vector) or vector of 1-character strings indicating the type of plot for each polygons, see <a href="#">graphics::matplot</a> for all possible types.
<code>xlab</code>	titles for x axe, as in <a href="#">graphics::matplot</a> .
<code>ylab</code>	titles for y axe, as in <a href="#">graphics::matplot</a> .
<code>ylim</code>	ranges of y axe, as in <a href="#">graphics::matplot</a> .
<code>\dots</code>	arguments passed to or from other methods.

## Value

No return value, called for side effects

## Examples

```
data(usa)
precarto <- precartogramR(usa, method="gsm", pf=1.2, verbose=TRUE)
plot(precarto)
```

---

```
precartogramR
```

*Make a pre cartogram analysis*

---

### Description

Make a pre cartogram analysis

### Usage

```
precartogramR(
  data,
  method = c("gsm", "gn", "dcn", "GastnerSeguyMore", "GastnerNewman",
    "DougenikChrismanNiemeyer"),
  gridpower2 = 8:11,
  pf = 1.5,
  verbose = FALSE
)
```

### Arguments

data	a sf object which contains at least two columns: obviously a geometry column (giving the map) and a column which contains a count by region (leading to a density by region, density to be equalized by deformation). Each row of data is a region and contains the simple feature geometry of type POLYGON or MULTIPOLYGON. Polygon ring directions are not checked but exterior ring must counter clockwise and holes clockwise (use option <code>check_ring_dir</code> of <code>sf::st_read</code> to achieve the right orientation of ring direction on import or use <code>check_ring_dir</code> function)
method	the method to be used, can be one of the following: <code>gsm</code> or <code>GastnerSeguyMore</code> (default), <code>gn</code> or <code>GastnerNewman</code> , <code>dcn</code> or <code>DougenikChrismanNiemeyer</code> .
gridpower2	a vector of exponent (to be raised at the power of 2) that gives the $\log_2(\text{size})$ of the grid (default to 8:11); meaningful for method <code>gsm</code> or <code>GastnerSeguyMore</code> (default), <code>gn</code> or <code>GastnerNewman</code>
pf	Determines space between map and boundary (default to 1.5); meaningful for method <code>gsm</code> or <code>GastnerSeguyMore</code> (default), <code>gn</code> or <code>GastnerNewman</code>
verbose	a boolean object to set on verbose mode (default to FALSE); meaningful for method <code>gsm</code> or <code>GastnerSeguyMore</code> (default), <code>gn</code> or <code>GastnerNewman</code>

### Value

either a `dbv.cartogramR` object (if method is `dcn` or `DougenikChrismanNiemeyer`) see [dist\\_between\\_vertices](#) for details or a `gridanalysis.cartogramR` (if method is `gsm` or `GastnerSeguyMore` (default), `gn` or `GastnerNewman`) see [grid\\_analysis](#) for details



## References

- Dougenik, J., Chrisman, R. & Niemeyer, D. (1985). An algorithm to construct continuous area cartograms. *Professional Geographer* **37**: 75-81.
- Gastner, M. & Newman, M.E.J. (2004). Diffusion-based method for producing density equalizing maps. *Proc. Natl. Acad. Sci. USA*, **101**:7499-7504
- Gastner, M., Seguy, V. & More, P. (2018). Fast flow-based algorithm for creating density-equalizing map projections. *Proceedings of the National Academy of Sciences USA*, **115**:E2156-E2164

## Examples

```
data(usa)
precarto <- precartogramR(usa)
plot(precarto)
summary(precarto)
```

---

print.cartogramR      *Print a cartogram object*

---

## Description

Print a cartogram object

## Usage

```
## S3 method for class 'cartogramR'
print(x, ...)
```

## Arguments

x                    a cartogramR object  
\dots                arguments passed to or from other methods.

## Value

No return value, called for side effects

---

```
print.summary.cartogramR
```

*Print a summary of a cartogram object*

---

### Description

Print a summary of a cartogram object

### Usage

```
## S3 method for class 'summary.cartogramR'  
print(x, ...)
```

### Arguments

x	a summary.cartogramR object
\dots	arguments passed to or from other methods. The following argument is available at this level : digits, the number of significant digits to use when printing.

### Value

x.

### Examples

```
data(usa)  
carto <- cartogramR(usa, "electors64")  
summary(carto)
```

---

```
residuals.cartogramR
```

*Errors of a cartogram object*

---

### Description

Errors of a cartogram object

### Usage

```
## S3 method for class 'cartogramR'  
residuals(object, ...)
```

**Arguments**

`object` a cartogramR object

`\dots` arguments passed to or from other methods. The following arguments are available: - `type`; a character string giving the type of residuals (see details) - "relative error" - "absolute error" - "symmetric difference" - `center` ; a character string giving the type of center: - "point\_on\_surface" (`st_point_on_surface` applied on original and on deformed/cartogram region). - "deformed\_center" (the center function, see `cartogramR_options`, is applied on region and this center follows the deformation giving the center on the deformed/cartogram region) - "centroid" (centroid of original and deformed/cartogram region). Can be abbreviated.

**Details**

The error vector contains the values of the differences between actual area of regions in the cartogram and theoretical area (obtained with conservation of total area and constant density over region in the final cartogram)

Relative error are the error vector divided by the theoretical area

Symmetric difference are the symmetric difference between actual area of regions in the cartogram and the original area. Each region is scaled to have an area equal to 1 and centered around the chosen center.

**Value**

A numeric vector which contains for each region observed area minus theoretical area

**Examples**

```
data(usa)
carto <- cartogramR(usa, "electors64")
residuals(carto)
```

---

summary.cartogramR      *Summary of a cartogram object*

---

**Description**

Summary of a cartogram object

**Usage**

```
## S3 method for class 'cartogramR'
summary(object, ...)
```

**Arguments**

object	a cartogramR object
\dots	arguments passed to or from other methods. The following arguments are available: - digits integer, used for number formatting with signif if not specified (i.e., [missing](.), [signif]() will not be called anymore (since \R >= 3.4.0, where the default has been changed to only round in the print and format methods). - quantile.type integer code used in quantile(*, type=quantile.type). - center character string code used in <a href="#">residuals.cartogramR</a> .

**Value**

A summary.cartogramR object: a list with the following components:

- qrr, the summary of absolute relative residuals
- qres, the summary of absolute residuals
- qsymdiff, the summary of all pairwise symmetric difference between two scaled (multi)polygons representative of two regions.

**Examples**

```
data(usa)
carto <- cartogramR(usa, "electors64")
summary(carto)
```

---

```
summary.dbv.cartogramR
```

*Summary of a dbv.cartogram object*

---

**Description**

Summary of a dbv.cartogram object

**Usage**

```
## S3 method for class 'dbv.cartogramR'
summary(object, ...)
```

**Arguments**

object            a dbv.cartogramR object  
\dots            arguments passed to or from other methods.

**Value**

a data-table which contains by region (L3)

- the sample quantiles corresponding to the probability 0.8, 0.85, ...,1
- the total number of vertices divided by the perimeter of the region (the sum of all polygons perimeter of the region, NbyPerim)

**Examples**

```
data(usa)
dbv <- dist_between_vertices(data=usa)
summary(dbv)
```

---

```
summary.gridanalysis.cartogramR
```

*Summary of a gridanalysis.cartogram object*

---

**Description**

Summary of a gridanalysis.cartogram object

**Usage**

```
## S3 method for class 'gridanalysis.cartogramR'
summary(object, ...)
```

**Arguments**

object            a gridanalysis.cartogramR object  
\dots            arguments passed to or from other methods.

**Value**

A vector which indicate the grid size necessary to have more than steps grid points in each polygon

**Examples**

```
data(usa)
ga <- grid_analysis(data=usa, gridpower2=4:9)
summary(ga)
```

---

to_coord_polygon	<i>Transform from coordinates system used in flow based cartogram to coordinates system used in the polygons</i>
------------------	--

---

**Description**

Apply the mapping from the coordinates system used in flow based cartogram to the coordinates system used in the polygons (characterised by the CRS)

**Usage**

```
to_coord_polygon(coord, carto)
```

**Arguments**

coord	a vector of length 2 or a two columns matrix containing xy coordinates to transform
carto	a cartogramR object

**Value**

a vector of length 2 or a two columns matrix containing xy coordinates in the coordinate systems of polygons used to build the cartogram

**Examples**

```
data(usa)
carto <- cartogramR(usa, "electors64")
to_coord_polygon(c(256,256), carto)
```

---

usa	<i>Map of the number of electors in each state of the USA</i>
-----	---

---

**Description**

This data set is a basemap of the conterminous USA with the the number of electors in the 49 states from 1964 to 2020.

**Usage**

```
data(usa)
```

**Format**

A sf object containing 24 columns of data and the geometry. Projected CRS: US National Atlas Equal Area (EPSG:2163)

# Index

## \* datasets

france\_dept, 9  
france\_rivers, 10  
usa, 22

as.sf, 2  
as.sf.cartogramR, 3  
as.sfc, 3  
as.sfc.cartogramR, 4  
as.sfmultipolygon, 4

cartogramR, 5, 6, 7  
cartogramR\_options, 5, 6, 6, 13, 19  
check\_ring\_dir, 5, 8, 16

dist\_between\_vertices, 9, 16

france\_dept, 9  
france\_rivers, 10  
from\_coord\_polygon, 10

geom\_cartogramR, 11  
graphics::matplot, 15  
grid\_analysis, 12, 16

make\_layer, 12

par, 14  
plot.cartogramR, 13  
plot.dbv.cartogramR, 14  
plot.gridanalysis.cartogramR, 15  
precartogramR, 16  
print.cartogramR, 17  
print.summary.cartogramR, 18

residuals.cartogramR, 18, 20

sf::st\_centroid, 7, 13  
sf::st\_point\_on\_surface, 7  
sf::st\_read, 5, 16  
st\_point\_on\_surface, 5, 19

summary.cartogramR, 19  
summary.dbv.cartogramR, 20  
summary.gridanalysis.cartogramR, 21

to\_coord\_polygon, 22

usa, 22